920476-904943

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

| | | |
|---|---|---|
| **In the application of** | : | Taylor, Graham |
| **Serial No.** | : | 09/981,444 |
| **Filed** | : | October 17, 2001 |
| **For** | : | Adaptive Software Interface |
| **Examiner** | : | Choudhury, Azizul Q. |
| **Art Unit** | : | 2145 |
| **Customer number** | : | 23644 |

## APPEAL BRIEF

Honorable Director of Patents and Trademarks
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

This appeal is from the Notice of Panel Decision From Pre-Appeal Brief Review mailed July 5, 2006 in which all pending claims (namely Claims 1 to 10, 18 to 20, and 22 to 26) were rejected. A timely Notice of Appeal was filed with the required fee.

This brief is being filed along with the required $500 fee pursuant to 37 C. F. R. § 41.20(b)(2), which should be deducted from Deposit Account No. 12-0913.

(i) Real Party in Interest

This application is assigned to Nortel Networks Limited who is the real party in interest.

(ii) Related Appeals and Interferences

There are no related appeals or interferences.

(iii) Status of Claims

This application was filed with claims 1 to 26. In the responses of April 5, 2005 and September 7, 2005 claims 1 to 10, 18, 19, 22, 24 & 25 were retained as originally filed, claims 20, 23 & 26 were amended and claims 11 to 17 & 21 cancelled. No claims have been allowed. Claims 1 to 10, 18 to 20, and 22 to 26 are those claims being appealed, and are set forth in the Claims Appendix.

(iv) Status of Amendments

A response was filed in connection with the final Office Action mailed July 8, 2005 and entered by the Examiner, so the claims now pending have all been considered by the Examiner and finally rejected. It is the rejection of these claims as set forth in the final Office Action mailed July 8, 2005 that is appealed.

(v) Summary of Claimed Subject Matter

In a first main aspect, the invention as presently claimed is concerned with a method of generating an adaptive software interface for at least two networked entities. The method comprises generating structured meta-data providing at least one semantic information element describing a characteristic of each said entity; collating the semantic information elements of each said entity where possible with corresponding semantic information elements of said at least one other entity; and analysing said collated semantic information elements to establish the extent to which the interface capabilities of said at least two networked entities are compatible and generating in accordance with said established compatibility the adaptive software interface for the two entities.

The invention concerns networked hardware and/or software applications and/or components which need to establish an appropriate interface for communicating with other hardware and/or software applications and/or components across the network. For example, in a communications or in a computer network, it is important for various network elements to be able to communicate and cooperate when running differing applications. This requires the capability for each application to establish an appropriate interface with other applications.

The invention is directed to an adaptive software interface which is able to mediate between different versions of interface definitions (see page 1, lines 7-9 of the present application). The problem addressed by the invention is that when using conventional IDLs (Interface Definition Languages) to generate software interfaces, different versions of interface definitions may use a different format for meta data which is used to describe interface characteristics. A problem is that entities having interfaces defined using different IDLs may appear to be wholly or partially incompatible with each other because of differences in format, whereas there

actually is some compatibility between the interfaces - see pages 1 and 2 of the description of the present application.

The solution of the present invention, as presented in claim 1, is to use structured meta data providing at least one semantic information element describing the characteristic of each of at least two entities, to collate and analyze the semantic information elements to establish the extent to which the interface capabilities of the entities are compatible, and to generate an adaptive software interface in accordance with the established compatibility. By using meta data providing at least one semantic element (as opposed to a purely syntactic information element) the meaning of characteristics can be used to determine compatibility rather than the format or syntax of any IDL used to generate the meta data. Furthermore, by generating an adaptive software interface component with the established compatibility, it is possible for two entities to communicate despite having interface definitions (i.e. meta data) defined using different formats.

In another aspect, the invention provides a method of determining at least one behavioural characteristic of a first entity in a relationship with at least one other entity. The method comprises the steps of: generating meta-data providing a structure containing at least one semantic information element describing a characteristic of the first entity; generating meta-data providing a structure containing at least one semantic information element describing a characteristic of the at least one other entity; collating the semantic information elements of the first entity with the semantic information elements of the at least one other entity; and analysing each pair of collated semantic information elements to determine at least one behavioural characteristic of the first entity in the relationship.

In yet another aspect, the invention provide a method of establishing a compatible interface between an initiator and a responder in the case where an interface of the initiator has at least one differing characteristic from an interface of the responder.

The method comprises the steps of generating at least one meta-data structure providing at least one semantic information element for each entity, wherein each said semantic information element describes a characteristic of an interface capability of one of said entities; collating said meta-data structures such that each semantic information element corresponding to the initiator's interface capability is collated with a corresponding semantic information element corresponding the responder's interface capability; analysing the collated semantic information elements to determine the extent to which the initiator and the responder can generate a compatible interface; and establishing in accordance with said analysis an interface between said initiator and said responder.

In other aspects, the invention provides a network management system, a program for a computer, a software application and a network related to the foregoing method aspects of the invention.

(vi) Grounds of Rejection To Be Reviewed on Appeal

There is one rejection at issue:

1. the rejection of claims 1 to 16, 18 to 20 and 22 to 26 under 35 USC §102(b) as being anticipated by Sun's "Jini Specifications Archive – v1.0" (referred to hereinafter as 'Jini').

(vii) Argument

Ground 1:

It has been stated during the examination procedure that claim 1 is anticipated by Jini. The basis for the 35 USC §102(b) rejection of claim 1 is that Jini discloses:

A method of generating an adaptive software interface for at least two networked entities, the method comprising:

generating structured meta-data providing at least one semantic information element describing a characteristic of each said entity **[see section LU 1.2 "Attributes", lines 1 to 7]**;

collating the semantic information elements of each said entity where possible with corresponding semantic information elements of said at least one other entity **[see section AR 2.1.2 "Lookup Service", lines 1 to 4]**; and

analysing said collated semantic information elements to establish the extent to which the interface capabilities of said at least two networked entities are compatible and generating in accordance with said established compatibility the adaptive software interface for the two entities **[see section AR 2.1.2 "Lookup Service", lines 1 to 10]**.

To demonstrate anticipation under 35 U.S.C. 102, a prior art reference must disclose each and every feature of an invention whether explicitly or inherently (see Hasani vs International Trade Comm'n, 126 F.3d 1473, 1477, 44 USPQ2d 1358, 1361 (Fed Cir 1997)).

It has been stated during the examination procedure (see "Response to Remarks" section, second paragraph of Office Action mailed July 8th, 2005) in response to Appellant's detailed reasons why Jini does not disclose the features of the present claims that "*It is important to not simply accept the literal translation of a prior art but,*

*to attain an understanding of the spirit of the design disclosed within a prior art. It is then that one is able to discern how each and every feature of the claimed invention is present within the prior art design*". Appellant was not aware that a new test for novelty had been established whereby it is no longer necessary to demonstrate that a prior art reference teaches each and every claim limitation, but that it is sufficient for an Examiner to attain enlightenment in the presence of the prior art reference thereby negating the need to meet the test for novelty established by Hasani and as required by 35 USC §102(b).

Appellant places its faith in the Hasani test and fails to see what the "*spirit of the design disclosed within a prior art*" reference has to do with demonstrating anticipation in accordance with 35 USC §102(b).

Referring again to claim 1, Jini, LU1.2, Attributes, lines 1 to 7 states "*The attributes of a service item are represented as a set of attribute sets. An individual attribute set is represented as an instance of some class for the Java platform, each attribute being a public field of that class. The class provides strong typing of both the set and the individual attributes. A service item can contain multiple instances of the same class with different attribute values, as well as multiple instances of different classes. For example, an item might have multiple instances of a Name class, each giving the common name of the service in a different language, plus an instance of a Location class, an Owner class, and various service-specific classes. The schema used for attributes is not constrained by this specification, but a standard foundation schema for Jini systems is defined in the Jini™ Lookup Attribute Schema Specification.*"

Appellant fails to see how this section of Jini can be construed to disclose "*generating structured meta-data providing at least one semantic information element describing a characteristic of each said entity*" and no proper reasoned statement that it does has been offered to date save for the fact that the Examiner "*is*

*convinced*" that it does and the need for the Appellant to somehow determine the "spirit of the design" disclosed by Jini.

Jini, AR 2.1.2, Lookup Service, lines 1 to 4 states "*Services are found and resolved by a* lookup service. *The lookup service is the central bootstrapping mechanism for the system and provides the major point of contact between the system and users of the system. In precise terms, a lookup service maps interfaces indicating the functionality provided by a service to sets of objects that implement the service. In addition, descriptive entries associated with a service allow more fine-grained selection of services based on properties understandable to people.*".

Appellant fails to see how this section of Jini can be construed to disclose "*collating the semantic information elements of each said entity where possible with corresponding semantic information elements of said at least one other entity*".

Jini, AR 2.1.2, Lookup Service, lines 1 to 10 states that "*Services are found and resolved by a* lookup service. *The lookup service is the central bootstrapping mechanism for the system and provides the major point of contact between the system and users of the system. In precise terms, a lookup service maps interfaces indicating the functionality provided by a service to sets of objects that implement the service. In addition, descriptive entries associated with a service allow more fine-grained selection of services based on properties understandable to people.*

*Objects in a lookup service may include other lookup services; this provides hierarchical lookup. Further, a lookup service may contain objects that encapsulate other naming or directory services, providing a way for bridges to be built between a Jini Lookup service and other forms of lookup service. Of course, references to a Jini Lookup service may be placed in these other naming and directory services, providing a means for clients of those services to gain access to a Jini system.*"

Appellant fails to see how this section of Jini can be construed to disclose "*analysing said collated semantic information elements to establish the extent to which the interface capabilities of said at least two networked entities are compatible and generating in accordance with said established compatibility the adaptive software interface for the two entities*".

In Jini, a service is not an interface (see Jini, AR 2.1.1, Services). A service is something that can be used by a person, a program, or another service. It can be computational, storage, a communication channel to another user, or another service. Examples of services include devices such as printers, displays, disks, software (such as applications or utilities), information (such as databases and files), and users of the system. Services will appear programmatically as objects in the Java programming language, perhaps made up of other objects in the Java programming language. A service will have an interface, which defines the operations that can be requested of that service. The type of the service determines the interfaces that make up that service (see Jini Technology Glossary). Even if it could be construed that a service is somehow also an interface, services are not adaptively generated in Jini, but are added or withdrawn through a registration process.

Whilst services in Jini have interfaces, there is no disclosure in the Jini specifications that such interfaces are adaptive having being generated after "*analysing said collated semantic information elements to establish the extent to which the interface capabilities of said at least two networked entities are compatible and generating [the adaptive software interface for the two entities] in accordance with said established compatibility*". For this reason alone, anticipation of claim 1 cannot be demonstrated based on Jini.

The present invention is directed to adaptive software interfaces as its title suggests. In other words it is directed to a software interface (i.e. software allowing

independent entities to act on or communicate with each other) which is adaptive (i.e. is generated to or adapted to the particular requirements of the entities it interfaces between). In particular, the present invention is directed to generating such a software interface on the basis of an analysis of the extent to which the interface capabilities of the entities are compatible. Different entities to be interfaced may be compatible (or incompatible, from another point of view) to different extents. The present invention enables a software interface to be generated which takes into account the extent of compatibility. This is achieved by generating, collating and analyzing semantic information about the entities to be interfaced and using that to generate the software interface.

The prior art Jini Lookup Service Specification is not related to generating adaptive software interfaces but to a particular service in a distributed computing platform known as Jini. Members of a Jini system are federated in order to share access to various services. A Jini system is not a set of clients and servers, or users and programs, or even programs and files. Instead, a Jini system consists of services that can be collected together for the performance of a particular task. Central to the Jini platform is the Jini Lookup Service which enables members to lookup (i.e. to identify and locate) suitable services for the performance of a particular task. The Jini Lookup service is the "services Yellow pages" of the Jini platform.

Generally, it can be seen that the Jini Lookup Service specification is not directly related to the present invention. More specifically, the Jini Lookup Service Specification also fails to disclose at least the following two features of the invention as claimed in claim 1.

1) Regarding the claimed feature "*analyzing collated semantic information elements to establish the extent to which the interface capabilities of at least two network entities are compatible*", it has been stated in the Advisory Action mailed October 13th, 2005 that:

"*The lookup service [of Jini] maps interfaces indicating the functionality provided by a service to sets of objects that implement the service. It also states that descriptive entries associated with a service allow a more fine-grained selection of services based on properties understandable to people. Hence mapping is performed to provide the appropriate service and descriptive entries can be assessed to better select a service.*"

However, Appellants point out that there is no disclosure of analysis to establish the extent to which the interface capabilities of two network entities are compatible. In the Jini art, there is inherently a member requiring a particular service, a number of services listed by the lookup service, and the lookup service itself. However, none of these are analyzed to determine compatibility of their interfaces. A listed service is merely identified on the basis of a description of the functionality required by a member. There is no teaching whatsoever about interfaces of members or services – i.e. how a member might actually act on or communicate with a service. Moreover, there is no teaching whatsoever about the compatibility or otherwise of the interfaces of services or members. The Jini art is simply not concerned with interface compatibility between networked entities.

2) Regarding the claimed feature "*generating the adaptive software interface in accordance with the established compatibility*", it has been stated in the Advisory Action that:

"*The Jini specification teaches that the lookup service maps interfaces ... that implement the service (lines 1-7). It is inherent that since a service is being mapped to and is being searched for, that when it is mapped it will be generated.*"

However, Appellants disagree. The Jini lookup service lists existing and available services in the Jini system, not services that have not even been generated yet. That

would be pointless. Furthermore, even if the Examiner were correct, which is denied, the Examiner seems to have overlooked that the claim feature requires that what is generated is a software interface not a service. There is no teaching whatsoever of generating software interfaces in the Jini art which is simply not concerned with enabling software interfaces to be generated which take into account the extent of compatibility between network entities.

In short, the Jini art is irrelevant to the subject matter of claim 1.

Similar comments apply to the remaining claims. Hence, Appellants believe that the Examiner has failed to substantiate the rejection of the claims as presented in the Claims Appendix under 35 USC §102(b).

Reversal of the Examiner is therefore clearly in order and is solicited.

August 4, 2006

Respectfully submitted,

Peter J. Shakula
Registration No. 40,808
Barnes & Thornburg LLP
P.O. Box 2786
Chicago, Illinois 60690-2786
(312) 214-4813
(312) 759-5646 (fax)

## Claims Appendix

1. A method of generating an adaptive software interface for at least two networked entities, the method comprising:

generating structured meta-data providing at least one semantic information element describing a characteristic of each said entity;

collating the semantic information elements of each said entity where possible with corresponding semantic information elements of said at least one other entity; and

analysing said collated semantic information elements to establish the extent to which the interface capabilities of said at least two networked entities are compatible and generating in accordance with said established compatibility the adaptive software interface for the two entities.

2. A method as claimed in claim 1, wherein the step of collating occurs dynamically during a preliminary exchange between the two entities prior to an interface being established between the two entities.

3. A method as claimed in claim 1, wherein said structured meta-data includes associated meta-data for at least one said semantic information element.

4. A method as claimed in claim 1, wherein the semantic information element describing the characteristics of said adaptive interface is provided in said meta-data in a form independent of the version of software used to generate said metadata.

5. A method as claimed in claim 1, wherein said semantic information element is generated by a compiler receiving input data from an interface description and a code template.

6. A method as claimed in claim 1, wherein said interface description includes a model of the data to be communicated across the interface and a code template.

7. A method as claimed in claim 1, wherein said semantic information element provided by said meta-data has a form which can be mapped to an appropriate transport layer and exchanged between said networked entities prior to a higher level interface being established between said networked entities.

8. A method of determining at least one behavioural characteristic of a first entity in a relationship with at least one other entity comprising the steps of:

generating meta-data providing a structure containing at least one semantic information element describing a characteristic of the first entity;

generating meta-data providing a structure containing at least one semantic information element describing a characteristic of the at least one other entity;

collating the semantic information elements of the first entity with the semantic information elements of the at least one other entity;

analysing each pair of collated semantic information elements to determine at least one behavioural characteristic of the first entity in the relationship.

9. A method as claimed in claim 8, wherein the meta-data structure is provided in a form suitable for indicating at least one semantic element taken from the group including: a description, a range, a default value.

10. A method as claimed in claim 8, wherein in the step of generating meta-data for the first entity, the at least one characteristic is a characteristic of an interface of the entity, and wherein in the step of generating meta-data for the at least one other entity, the at least one characteristic is a characteristic of an interface of the at least one other entity.

18. A method of establishing a compatible interface between an initiator and a responder in the case where an interface of the initiator has at least one differing characteristic from an interface of the responder comprising the steps of

generating at least one meta-data structure providing at least one semantic information element for each entity, wherein each said semantic information element describes a characteristic of an interface capability of one of said entities;

collating said meta-data structures such that each semantic information element corresponding to the initiator's interface capability is collated with a corresponding semantic information element corresponding the responder's interface capability;

analysing the collated semantic information elements to determine the extent to which the initiator and the responder can generate a compatible interface;

establishing in accordance with said analysis an interface between said initiator and said responder.

19. A network management system adapted to perform the steps in a method of generating an adaptive software interface for at least two entities, the method comprising:

generating structured meta-data providing at least one semantic information element describing a characteristic of each said entity;

collating the semantic information elements of each said entity with those stored semantic information elements of said at least one other entity; and

analysing said collated semantic information elements to establish the extent to which the interface capabilities of said at least two networked entities are compatible and generating in accordance with said established compatibility the adaptive software interface for the two entities.

20. A program for a computer stored on a computer-readable medium and arranged to perform steps in a method of generating an adaptive software interface for at least two networked entities, the method comprising:

generating structured meta-data providing at least one semantic information element describing a characteristic of each said entity;

collating the semantic information elements of each said entity with those semantic information elements of said at least one other entity; and

analysing said collated semantic information elements to establish the extent to which the interface capabilities of said at least two networked entities are compatible and generating in accordance with said established compatibility the adaptive software interface for the two entities.

22. A network including a computer system adapted to perform steps in a method of generating an adaptive software interface for at least two networked entities, the method comprising:

generating structured meta-data providing at least one semantic information element describing a characteristic of each said entity;

collating the semantic information elements of each said entity with those semantic information elements of said at least one other entity; and

analysing said collated semantic information elements to establish the extent to which the interface capabilities of said at least two networked entities are compatible and generating in accordance with said established compatibility the adaptive software interface for the two entities.

23. A software application stored on a computer-readable medium and capable of providing a semantic description of another application performing a computational process in a network, the semantic description having a predetermined structure which is invariant regarding the version of compiler used to generate said semantic description from said software application and said other application, said semantic description providing discernable features of at least one characteristic of said other application.

24. A software application as claimed in claim 23, wherein said network is taken from the group comprising:

a communications network, a data network, a computer network.

25. A software application as claimed in claim 23, wherein said at least one characteristic relates to a characteristic of an ability of said other application to interface with at least one other application performing a computational process over said network.

26. An adaptive software interface for at least two networked entities, the adaptive software interface being stored on a computer-readable medium, the adaptive software interface being generated by;

generating structured meta-data providing at least one semantic information element describing a characteristic of each said entity;

collating the semantic information elements of each said entity where possible with corresponding semantic information elements of said at least one other entity; and

analysing said collated semantic information elements to establish the extent to which the interface capabilities of said at least two networked entities are compatible and generating in accordance with said established compatibility the adaptive software interface for the two entities.

## Evidence Appendix and Related Proceedings Appendix

There are no such appendices.